AMENDMENTS TO THE CLAIMS

Please cancel claims 1-20, 23-29, 32, and 34 without prejudice. The listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1-20. (canceled)

21. (previously presented) An apparatus for speculatively forwarding storehit data in a microprocessor pipeline, the apparatus comprising:

first and second virtual address comparators, for comparing a virtual load address with first and second virtual store addresses to generate a virtual match signal for indicating whether first and second storehit data is likely present in a store buffer and a result forwarding cache, respectively, of the microprocessor, wherein if said first and second storehit data are both present said second storehit data is newer than said first storehit data;

first and second physical address comparators, for comparing a physical load address translated from said virtual load address with first and second physical store addresses translated from said plurality of virtual store addresses to generate a physical match signal for indicating whether said first and second storehit data is certainly present in said store buffer and said result forwarding cache, respectively;

forwarding logic, coupled to receive said virtual match signal, for forwarding said second storehit data present in said store buffer in response to said virtual match signal indicating no match between said virtual load address and said second virtual store addresses but a match between said virtual load address and said first virtual store address, prior to generation of said physical match signal; and

control logic, for receiving said virtual and physical match signals and generating a stall signal for stalling the pipeline subsequent to said forwarding logic forwarding said storehit data from said store buffer if said physical match

signal indicates a match between said physical load address and said second physical store addresses although said virtual match signal previously indicated no match between said virtual load address and said second virtual store addresses, until correct data specified by said physical load address is provided to replace said previously forwarded second storehit data.

22. (previously presented) The apparatus of claim 21, further comprising:

a data unit, configured to forward said correct data specified by said physical load address to replace said previously forwarded second storehit data;

wherein said control logic is configured to deassert said stall signal after said data unit forwards said correct data.

23-29. (canceled)

30. (previously presented) A method for speculatively forwarding storehit data in a microprocessor pipeline, the method comprising:

determining that a virtual load address matches a first virtual store addresses present in the pipeline to indicate first storehit data is likely present in a store buffer of the microprocessor, but does not match a second virtual store address present in the pipeline to indicate second newer storehit data is likely absent in a result forwarding cache of the microprocessor;

forwarding the first storehit data from a first stage comprising the store buffer to a second stage of the pipeline having a load instruction specifying the load address based on said determining that the first storehit data is likely present in the store buffer and the second newer storehit data is likely absent in the result forwarding cache;

detecting that a physical load address translated from said virtual load address matches a physical store address translated from the second virtual store address to indicate the second newer storehit data is certainly present in the result forwarding cache , subsequent to said forwarding the first storehit data; and

stalling the pipeline in response to said detecting that said physical load address translated from said virtual load address matches said physical store address present in the pipeline, until correct data specified by said physical load address is provided to replace the previously forwarded second newer storehit data.

31. (original) The method of claim 30, further comprising:

forwarding correction data from a third stage of the pipeline to said second stage after said stalling the pipeline; and

unstalling the pipeline after said forwarding said correction data.

32. (canceled)

33. (original) The method of claim 30, wherein said storehit data comprises a store instruction result within the pipeline having an identical physical store address as said physical load address.

34. (canceled)

35. (previously presented) A method for speculatively forwarding storehit data in a microprocessor pipeline, the method comprising:

comparing a virtual load address with first and second virtual store addresses, wherein a load instruction specifying the virtual load address is newer than a first store instruction specifying the first virtual store address, which is newer than a second store instruction specifying the second virtual store address;

speculatively forwarding a result of the first store instruction to the load instruction, in response to said comparing indicating the virtual load address matches the first virtual store address and mismatches the second virtual store address;

comparing a physical load address with a physical store address, wherein the physical load address is a translation of the virtual load address, wherein

the physical store address is a translation of the second virtual store address;

determining said forwarding the result of the first store instruction to the load instruction was incorrect, after said speculatively forwarding the result of the first store instruction, in response to said comparing indicating the physical load address matches the physical store address; and

forwarding a result of the second store instruction to the load instruction, in response to said determining.

36. (previously presented) The method of claim 35, wherein said comparing the virtual load address with the first virtual store address comprises determining whether the virtual load address hits in a data cache of the microprocessor, wherein said speculatively forwarding the result of the first store instruction to the load instruction comprises the data cache providing the result of the first store instruction to the load instruction.

37. (previously presented) The method of claim 35, further comprising:

stalling the pipeline, in response to said determining, until said forwarding the result of the second store instruction to the load instruction.

38. (previously presented) The method of claim 35, further comprising:

writing the result of the second store instruction to a data cache of the microprocessor, in response to said determining;

wherein said forwarding the result of the second store instruction to the load instruction comprises the data cache providing the result of the second store instruction to the load instruction.

39. (previously presented) The method of claim 35, further comprising:

reissuing the load instruction, in response to said determining.

40. (previously presented) The method of claim 39, wherein said reissuing the load instruction comprises providing the virtual load address to a data cache of the microprocessor from a replay buffer of the microprocessor.

41. (previously presented) The method of claim 35, wherein said comparing the virtual load address with the second virtual store address is performed prior to the result of the second store instruction being stored a store buffer of the microprocessor.

42. (previously presented) The method of claim 35, wherein said speculatively forwarding the result of the second store instruction to the load instruction comprises forwarding the result of the second store instruction from a result forwarding cache (RFC) of the microprocessor.

43. (previously presented) The method of claim 42, further comprising:

caching the result of the second store instruction in the RFC, prior to said comparing the virtual load address with the second virtual store address.

44. (previously presented) The method of claim 43, further comprising:

caching a result of a non-store instruction in the RFC, prior to said comparing the virtual load address with the second virtual store address.